UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/017,337	12/13/2001	Qiang Cao	M-12069 US	6779

34036 7590 03/25/2005

SILICON VALLEY PATENT GROUP LLP
2350 MISSION COLLEGE BOULEVARD
SUITE 360
SANTA CLARA, CA 95054

EXAMINER	
THANGAVELU, KANDASAMY	
ART UNIT	**PAPER NUMBER**
2123	

DATE MAILED: 03/25/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/017,337	CAO ET AL.	
	Examiner	Art Unit	
	Kandasamy Thangavelu	2123	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *13 December 2001*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *29* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-29* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *13 December 2001* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-29 of the application have been examined.

Drawings

2. The drawings submitted on December 13, 2001 are accepted.

Claim Rejections - 35 USC § 101

3. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

4. Claims 1-10 are rejected under 35 U.S.C. 101 because the claimed inventions are directed to non-statutory subject matter.

Method claims 1-10 are rejected for reciting a process that is not directed to the technological arts.

Regarding claim 1, this claim is directed at a method for simulating different mean time to recover (MTTR) settings, whereas none of the limitations describe any type of computer-implemented steps. To be statutory, the utility of an invention must be within the technological arts. *In re Musgrave*, 167 USPQ 280, 289-90 (CCPA, 1970). The definition of "technology" is the "application of science and engineering to the development of machines and procedures in

order to enhance or improve human conditions, or at least to improve human efficiency in some respect." (Computer Dictionary 384 (Microsoft Press, 2d ed. 1994)).

Dependent claims 2-10 depend on Claim 1 but do not add further statutory steps.

The limitations recited in claims 1-10 contain no language suggesting these claims are intended to be within the technological arts.

5. Claim 1- 10 would be statutory if claim 1is rewritten as a computer implemented method for simulating different mean time to recover (MTTR) settings, the method comprising ...

Claim Rejections - 35 USC § 103

6. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains.

7. The factual inquiries set forth in *Graham* **v.** *John Deere Co.*, 383 U.S. 1, 148 USPQ 459 (1966), that are applied for establishing a background for determining obviousness under 35 U.S.C. 103(a) are summarized as follows:

1. Determining the scope and contents of the prior art.
2. Ascertaining the differences between the prior art and the claims at issue.
3. Resolving the level of ordinary skill in the pertinent art.

4. Considering objective evidence present in the application indicating obviousness or nonobviousness.

8. Claims 1-2, 11-12 and 21-26 are rejected under 35 U.S.C. 103(a) as being unpatentable over **Lee et al.** ("Checkpointing schemes for fast restart in main memory database systems", IEEE, 1997) in view of **Nicola et al.** ("Comparative analysis of different models of checkpointing and recovery", IEEE 1990), and further in view of **Joshi et al.** ("Checkpointing in Oracle", Proceedings of the 24th VLDB conference, 1998).

8.1 **Lee et al.** teaches Checkpointing schemes for fast restart in main memory database systems. Specifically, as per claim 1, **Lee et al.** teaches a method for simulating and evaluating the performance of different checkpointing schemes for database systems (Abstract, L6-13; Page 665, CL1, Para 3, L1-3).

Lee et al. does not expressly teach a method for simulating different mean time to recover (MTTR) settings. **Nicola et al.** teaches a method for simulating different mean time to recover (MTTR) settings (Abstract, L3-8 and L11-12; Page 818, CL2, Para 2, L1-2; Page 820, CL1, Para 3 to Page 821, CL1, Para 1), because recovery times depend on the checkpointing strategy (Abstract, L11-12); and the objective is to minimize the time spent in checkpointing and recovery; increasing the frequency of checkpointing decreases recovery time but increases the checkpointing time; decreasing the frequency of checkpointing decreases the checkpointing time but increases the recovery time; there is an optimum checkpointing strategy which maximizes the system availability for processing new transactions and minimizes the mean response time of

a transaction (Page 807, CL2, Para 1, L12-23). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the method of **Lee et al.** with the method of **Nicola et al.** that included a method for simulating different mean time to recover (MTTR) settings. The artisan would have been motivated because recovery times would depend on the checkpointing strategy; and the objective would be to minimize the time spent in checkpointing and recovery; increasing the frequency of checkpointing would decrease recovery time but would increase the checkpointing time; decreasing the frequency of checkpointing would decrease the checkpointing time but would increase the recovery time; there would be an optimum checkpointing strategy which would maximize the system availability for processing new transactions and would minimize the mean response time of a transaction.

Lee et al. does not expressly teach providing a simulated checkpoint queue, the simulated checkpoint queue being associated with a simulated MTTR setting, the simulated checkpoint queue being an ordered list of one or more elements, each of the one or more elements representing a respective buffer, the ordered list having a head and a tail. **Joshi et al.** teaches providing a simulated checkpoint queue, the simulated checkpoint queue being associated with a simulated MTTR setting, the simulated checkpoint queue being an ordered list of one or more elements, each of the one or more elements representing a respective buffer, the ordered list having a head and a tail (Page 665, CL2, Para 3; Page 667, CL2, Para 3), because the ordered queues increase the efficiency of identifying the precise set of buffers that need to be written for checkpoints and provides scalability (Page 665, CL2, Para 3); such queues have a head and a tail; and the buffers are written from the head of the checkpoint queue (Page 667, CL2, Para 3). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to

modify the method of **Lee et al.** with the method of **Joshi et al.** that included providing a simulated checkpoint queue, the simulated checkpoint queue being associated with a simulated MTTR setting, the simulated checkpoint queue being an ordered list of one or more elements, each of the one or more elements representing a respective buffer, the ordered list having a head and a tail. The artisan would have been motivated because the ordered queues would increase the efficiency of identifying the precise set of buffers that would need to be written for checkpoints and would provide scalability; such queues would have a head and a tail; and the buffers would be written from the head of the checkpoint queue.

Lee et al. does not expressly teach in response to detecting a change to a first buffer in a normal checkpoint queue, checking if the first buffer is represented in the simulated checkpoint queue, and if the first buffer is not represented in the simulated checkpoint queue, linking an element that represents the first buffer to the tail of the simulated checkpoint queue. **Joshi et al.** teaches in response to detecting a change to a first buffer in a normal checkpoint queue, checking if the first buffer is represented in the simulated checkpoint queue, and if the first buffer is not represented in the simulated checkpoint queue, linking an element that represents the first buffer to the tail of the simulated checkpoint queue (Page 667, CL1, Para 3), because each such entry on the checkpointing queue contains the entry up to which the buffers need to be written in order to complete the checkpoint represented by the entry (Page 667, CL1, Para 3). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the method of **Lee et al.** with the method of **Joshi et al.** that included in response to detecting a change to a first buffer in a normal checkpoint queue, checking if the first buffer was represented in the simulated checkpoint queue, and if the first buffer was not represented in the simulated

checkpoint queue, linking an element that represented the first buffer to the tail of the simulated checkpoint queue. The artisan would have been motivated because each such entry on the checkpointing queue would contain the entry up to which the buffers would need to be written in order to complete the checkpoint represented by the entry.

8.2 As per claim 2, **Lee et al., Nicola et al.** and **Joshi et al.** teach the method of claim 1. **Lee et al.** does not expressly teach providing a simulated write counter, the simulated write counter being associated with the simulated MTTR setting; determining if linking the element to the tail of the simulated checkpoint queue causes the simulated checkpoint queue to exceed a predetermined length. **Nicola et al.** teaches providing a simulated write counter, the simulated write counter being associated with the simulated MTTR setting; determining if linking the element to the tail of the simulated checkpoint queue causes the simulated checkpoint queue to exceed a predetermined length (Abstract, L3-8 and L11-12), because recovery times depend on the checkpointing strategy (Abstract, L11-12); and the objective is to minimize the time spent in checkpointing and recovery; increasing the frequency of checkpointing (decreasing the checkpoint queue length) decreases recovery time but increases the checkpointing time; decreasing the frequency of checkpointing (increasing the checkpoint queue length) decreases the checkpointing time but increases the recovery time; there is an optimum checkpointing strategy which maximizes the system availability for processing new transactions and minimizes the mean response time of a transaction (Page 807, CL2, Para 1, L12-23). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the method of **Lee et al.** with the method of **Nicola et al.** that included providing a simulated write

counter, the simulated write counter being associated with the simulated MTTR setting; determining if linking the element to the tail of the simulated checkpoint queue would cause the simulated checkpoint queue to exceed a predetermined length. The artisan would have been motivated because recovery times would depend on the checkpointing strategy; and the objective would be to minimize the time spent in checkpointing and recovery; increasing the frequency of checkpointing (decreasing the checkpoint queue length) would decrease recovery time but would increase the checkpointing time; decreasing the frequency of checkpointing (increasing the checkpoint queue length) would decrease the checkpointing time but would increase the recovery time; there would be an optimum checkpointing strategy which would maximize the system availability for processing new transactions and would minimize the mean response time of a transaction.

Lee et al. does not expressly teach in response to determining that the simulated checkpoint queue exceeds the predetermined length, removing an element from the head of the simulated checkpoint queue and incrementing the simulated write counter. **Joshi et al.** teaches in response to determining that the simulated checkpoint queue exceeds the predetermined length, removing an element from the head of the simulated checkpoint queue and incrementing the simulated write counter (Page 667, CL2, Para 3), because checkpointing schemes write buffers from the head of the checkpoint queue, so the instance of checkpoint will keep advancing (Page 667, CL2, Para 3). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the method of **Lee et al.** with the method of **Joshi et al.** that included in response to determining that the simulated checkpoint queue exceeds the predetermined length, removing an element from the head of the simulated checkpoint queue and

incrementing the simulated write counter. The artisan would have been motivated because checkpointing schemes would write buffers from the head of the checkpoint queue, so the instance of checkpoint would keep advancing.

8.3 As per Claims 11 and 12, these are rejected based on the same reasoning as Claims 1 and 2, supra. Claims 11 and 12 are computer-readable storage medium claims reciting the same limitations as Claims 1 and 2, as taught throughout by **Lee et al., Nicola et al.** and **Joshi et al.**

8.4 As per claim 21, **Lee et al.** teaches a system comprising a memory; one or more processors coupled to the memory; and a simulated setting maintained in the memory (Abstract, L6-13; Page 665, CL1, Para 3, L1-3).

Lee et al. does not expressly teach a simulated MTTR setting maintained in the memory. **Nicola et al.** teaches a simulated MTTR setting maintained in the memory (Abstract, L3-8 and L11-12; Page 818, CL2, Para 2, L1-2; Page 820, CL1, Para 3 to Page 821, CL1, Para 1), because recovery times depend on the checkpointing strategy (Abstract, L11-12); and the objective is to minimize the time spent in checkpointing and recovery; increasing the frequency of checkpointing decreases recovery time but increases the checkpointing time; decreasing the frequency of checkpointing decreases the checkpointing time but increases the recovery time; there is an optimum checkpointing strategy which maximizes the system availability for processing new transactions and minimizes the mean response time of a transaction (Page 807, CL2, Para 1, L12-23). It would have been obvious to one of ordinary skill in the art at the time

of Applicants' invention to modify the system of **Lee et al.** with the system of **Nicola et al.** that included a simulated MTTR setting maintained in the memory. The artisan would have been motivated because recovery times would depend on the checkpointing strategy; and the objective would be to minimize the time spent in checkpointing and recovery; increasing the frequency of checkpointing would decrease recovery time but would increase the checkpointing time; decreasing the frequency of checkpointing would decrease the checkpointing time but would increase the recovery time; there would be an optimum checkpointing strategy which would maximize the system availability for processing new transactions and would minimize the mean response time of a transaction.

Lee et al. does not expressly teach providing a simulated checkpoint queue maintained in the memory, the simulated checkpoint queue being associated with a simulated MTTR setting, the simulated checkpoint queue being an ordered list of one or more elements, each of the one or more elements representing a respective buffer, the ordered list having a head and a tail. **Joshi et al.** teaches providing a simulated checkpoint queue maintained in the memory, the simulated checkpoint queue being associated with a simulated MTTR setting, the simulated checkpoint queue being an ordered list of one or more elements, each of the one or more elements representing a respective buffer, the ordered list having a head and a tail (Page 665, CL2, Para 3; Page 667, CL2, Para 3), because the ordered queues increase the efficiency of identifying the precise set of buffers that need to be written for checkpoints and provides scalability (Page 665, CL2, Para 3); such queues have a head and a tail; and the buffers are written from the head of the checkpoint queue (Page 667, CL2, Para 3). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the system of **Lee et al.** with the system

of **Joshi et al.** that included a simulated checkpoint queue maintained in the memory, the simulated checkpoint queue being associated with a simulated MTTR setting, the simulated checkpoint queue being an ordered list of one or more elements, each of the one or more elements representing a respective buffer, the ordered list having a head and a tail. The artisan would have been motivated because the ordered queues would increase the efficiency of identifying the precise set of buffers that would need to be written for checkpoints and would provide scalability; such queues would have a head and a tail; and the buffers would be written from the head of the checkpoint queue.

Lee et al. does not expressly teach a simulated write counter maintained in the memory, the simulated write counter being associated with the simulated MTTR setting. **Nicola et al.** teaches a simulated write counter maintained in the memory, the simulated write counter being associated with the simulated MTTR setting (Abstract, L3-8 and L11-12), because recovery times depend on the checkpointing strategy (Abstract, L11-12); and the objective is to minimize the time spent in checkpointing and recovery; increasing the frequency of checkpointing (decreasing the checkpoint queue length) decreases recovery time but increases the checkpointing time; decreasing the frequency of checkpointing (increasing the checkpoint queue length) decreases the checkpointing time but increases the recovery time; there is an optimum checkpointing strategy which maximizes the system availability for processing new transactions and minimizes the mean response time of a transaction (Page 807, CL2, Para 1, L12-23). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the system of **Lee et al.** with the system of **Nicola et al.** that included a simulated write counter maintained in the memory, the simulated write counter being associated with the

simulated MTTR setting. The artisan would have been motivated because recovery times would depend on the checkpointing strategy; and the objective would be to minimize the time spent in checkpointing and recovery; increasing the frequency of checkpointing (decreasing the checkpoint queue length) would decrease recovery time but would increase the checkpointing time; decreasing the frequency of checkpointing (increasing the checkpoint queue length) would decrease the checkpointing time but would increase the recovery time; there would be an optimum checkpointing strategy which would maximize the system availability for processing new transactions and would minimize the mean response time of a transaction.

Lee et al. does not expressly teach the simulated write counter providing a count of the number of times an element is removed from the simulated checkpoint queue. **Joshi et al.** teaches the simulated write counter providing a count of the number of times an element is removed from the simulated checkpoint queue (Page 666, CL2, Para 3 to Page 667, CL1, Para 1), because that allows checkpointing to record the checkpoint as having completed and update the checkpoint progress record (redo log) (Page 667, CL1, Para 1). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the system of **Lee et al.** with the system of **Joshi et al.** that included the simulated write counter providing a count of the number of times an element is removed from the simulated checkpoint queue. The artisan would have been motivated because that would allow checkpointing to record the checkpoint as having completed and update the checkpoint progress record (redo log).

Lee et al. does not expressly teach that the element is removed from the simulated checkpoint queue in response to a write out of a buffer from volatile memory and storing in nonvolatile memory. **Joshi et al.** teaches that the element is removed from the simulated

checkpoint queue in response to a write out of a buffer from volatile memory and storing in nonvolatile memory (Page 666, CL1, Para 3 to CL2, Para 1; Page 666, CL2, Para 2 to Page 667, CL1, Para 1), because after a buffer is written, it is unlinked from the checkpoint queue and the checkpointing records the checkpoint as having completed by updating the checkpoint progress record (Page 666, CL2, Para 3 to Page 667, CL1, Para 1). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the system of **Lee et al.** with the system of **Joshi et al.** that included the element being removed from the simulated checkpoint queue in response to a write out of a buffer from volatile memory and storing in nonvolatile memory. The artisan would have been motivated because after a buffer was written, it would be unlinked from the checkpoint queue and the checkpointing would record the checkpoint as having completed by updating the checkpoint progress record.

8.5 As per claim 22, **Lee et al., Nicola et al.** and **Joshi et al.** teach the system of claim 21. **Lee et al.** does not expressly teach that the element removed from the simulated checkpoint queue represents the buffer written out from volatile memory and stored in nonvolatile memory. **Joshi et al.** teaches that the element removed from the simulated checkpoint queue represents the buffer written out from volatile memory and stored in nonvolatile memory (Page 666, CL2, Para 3 to Page 667, CL1, Para 1), because after a buffer is written, it is unlinked from the checkpoint queue and the checkpointing records the checkpoint as having completed by updating the checkpoint progress record (Page 666, CL2, Para 3 to Page 667, CL1, Para 1). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the system of **Lee et al.** with the system of **Joshi et al.** that included the element removed from the

simulated checkpoint queue representing the buffer written out from volatile memory and stored in nonvolatile memory. The artisan would have been motivated because after a buffer was written, it would be unlinked from the checkpoint queue and the checkpointing would record the checkpoint as having completed by updating the checkpoint progress record.

8.6 As per claim 23, **Lee et al., Nicola et al.** and **Joshi et al.** teach the system of claim 22. **Lee et al.** teaches that the buffer is written out from volatile memory and stored in nonvolatile memory as a result of an incremental checkpoint operation involving the buffer (Page 663, CL1, Para 3, L6-8).

8.7 As per claim 24, **Lee et al., Nicola et al.** and **Joshi et al.** teach the system of claim 21. **Lee et al.** does not expressly teach that the buffer is written out from volatile memory to nonvolatile memory as a result of a second element being linked to the simulated checkpoint queue. **Joshi et al.** teaches that the buffer is written out from volatile memory to nonvolatile memory as a result of a second element being linked to the simulated checkpoint queue (Page 667, CL1, Para 3), because each such entry on the checkpointing queue contains the entry up to which the buffers need to be written in order to complete the checkpoint represented by the entry (Page 667, CL1, Para 3). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the system of **Lee et al.** with the system of **Joshi et al.** that included the buffer being written out from volatile memory to nonvolatile memory as a result of a second element being linked to the simulated checkpoint queue. The artisan would have been motivated because each such entry on the checkpointing queue would contain the

entry up to which the buffers would need to be written in order to complete the checkpoint represented by the entry.

Lee et al. does not expressly teach that linking the second element causes the simulated checkpoint queue to exceed a predetermined length. **Nicola et al.** teaches that linking the second element causes the simulated checkpoint queue to exceed a predetermined length (Abstract, L3-8 and L11-12), because recovery times depend on the checkpointing strategy (Abstract, L11-12); (Abstract, L11-12); and the objective is to minimize the time spent in checkpointing and recovery; increasing the frequency of checkpointing (decreasing the checkpoint queue length) decreases recovery time but increases the checkpointing time; decreasing the frequency of checkpointing (increasing the checkpoint queue length) decreases the checkpointing time but increases the recovery time; there is an optimum checkpointing strategy which maximizes the system availability for processing new transactions and minimizes the mean response time of a transaction (Page 807, CL2, Para 1, L12-23). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the system of **Lee et al.** with the system of **Nicola et al.** that included linking the second element causes the simulated checkpoint queue to exceed a predetermined length. The artisan would have been motivated because recovery times would depend on the checkpointing strategy; and the objective would be to minimize the time spent in checkpointing and recovery; increasing the frequency of checkpointing (decreasing the checkpoint queue length) would decrease recovery time but would increase the checkpointing time; decreasing the frequency of checkpointing (increasing the checkpoint queue length) would decrease the checkpointing time but would increase the recovery time; there would be an

optimum checkpointing strategy which would maximize the system availability for processing new transactions and would minimize the mean response time of a transaction.

8.8 As per claim 25, **Lee et al., Nicola et al.** and **Joshi et al.** teach the system of claim 24. **Lee et al.** does not expressly teach that the second element does not represent the buffer written out from volatile memory and stored in nonvolatile memory. **Joshi et al.** teaches that the second element does not represent the buffer written out from volatile memory and stored in nonvolatile memory (Fig. 1, buffers b_2, b_5, b_7), because clean buffers need not be written from volatile memory to nonvolatile memory (Fig. 1, buffers b_2, b_5, b_7). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the system of **Lee et al.** with the system of **Joshi et al.** that included the second element not representing the buffer written out from volatile memory and stored in nonvolatile memory. The artisan would have been motivated because clean buffers need not be written from volatile memory to nonvolatile memory.

8.8 As per claim 26, **Lee et al., Nicola et al.** and **Joshi et al.** teach the system of claim 24. **Lee et al.** does not expressly teach that the second element is linked to the simulated checkpoint queue in response to a modification to a second buffer, the second buffer being represented by the second element. **Joshi et al.** teaches that the second element is linked to the simulated checkpoint queue in response to a modification to a second buffer, the second buffer being represented by the second element (Page 667, CL1, Para 3), because each such entry on the checkpointing queue contains the entry up to which the buffers need to be written in order to

complete the checkpoint represented by the entry (Page 667, CL1, Para 3). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the system of **Lee et al.** with the system of **Joshi et al.** that included the second element being linked to the simulated checkpoint queue in response to a modification to a second buffer, the second buffer being represented by the second element. The artisan would have been motivated because each such entry on the checkpointing queue would contain the entry up to which the buffers would need to be written in order to complete the checkpoint represented by the entry.

9. Claims 3, 8, 13-18 and 27 are rejected under 35 U.S.C. 103(a) as being unpatentable over **Lee et al.** ("Checkpointing schemes for fast restart in main memory database system", IEEE, 1997) in view of **Nicola et al.** ("Comparative analysis of different models of checkpointing and recovery", IEEE 1990), and further in view of **Joshi et al.** ("Checkpointing in Oracle", Proceedings of the 24th VLDB conference, 1998) and **Schofield et al.** (U.S. Patent 6,493,826).

9.1 As per claim 3, **Lee et al., Nicola et al.** and **Joshi et al.** teach the method of claim 1. **Lee et al.** does not expressly teach that the predetermined length being a dirty buffer limit. **Joshi et al.** teaches a dirty buffer where data structures used for checkpointing are kept in the cache (Page 667, CL2, Para 3), because that allows checkpointing schemes to write buffers from the head of the checkpoint queue, so the instance of checkpoint keeps advancing (Page 667, CL2, Para 3). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the method of **Lee et al.** with the method of **Joshi et al.** that included a dirty buffer where data structures used for checkpointing were kept in the cache. The artisan would have

been motivated because that would allow checkpointing schemes to write buffers from the head of the checkpoint queue, so the instance of checkpoint would keep advancing.

Schofield et al. teaches the predetermined length being a dirty buffer limit (CL7, L60-64), because that allows writing the buffers and the logs to the disk (nonvolatile memory) when the information in the virtual storage reaches a predetermined size (CL7, L60-64). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the method of **Lee et al.** with the method of **Schofield et al.** that included the predetermined length being a dirty buffer limit. The artisan would have been motivated because that would allow writing the buffers and the logs to the disk (nonvolatile memory) when the information in the virtual storage reached a predetermined size.

9.2 As per claim 8, **Lee et al., Nicola et al.** and **Joshi et al.** teach the method of claim 1. **Lee et al.** does not expressly teach determining a dirty buffer limit for the simulated checkpoint queue, the dirty buffer limit specifying the length of the simulated checkpoint queue. **Joshi et al.** teaches a dirty buffer for the simulated checkpoint queue (Page 667, CL2, Para 3), because that allows checkpointing schemes to write buffers from the head of the checkpoint queue, so the instance of checkpoint keeps advancing (Page 667, CL2, Para 3). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the method of **Lee et al.** with the method of **Joshi et al.** that included a dirty buffer for the simulated checkpoint queue. The artisan would have been motivated because that would allow checkpointing schemes to write buffers from the head of the checkpoint queue, so the instance of checkpoint would keep advancing.

Schofield et al. teaches determining a dirty buffer limit for the simulated checkpoint queue, the dirty buffer limit specifying the length of the simulated checkpoint queue (CL7, L60-64), because that allows writing the buffers and the logs to the disk (nonvolatile memory) when the information in the virtual storage reaches a predetermined size (CL7, L60-64). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the method of **Lee et al.** with the method of **Schofield et al.** that included determining a dirty buffer limit for the simulated checkpoint queue, the dirty buffer limit specifying the length of the simulated checkpoint queue. The artisan would have been motivated because that would allow writing the buffers and the logs to the disk (nonvolatile memory) when the information in the virtual storage reached a predetermined size.

Lee et al. does not expressly teach the dirty buffer limit being determined from the simulated MTTR setting and historical operating data. **Nicola et al.** teaches the dirty buffer limit being determined from the simulated MTTR setting and historical operating data (Abstract, L3-8 and L11-12; Page 818, CL2, Para 2, L1-2; Page 820, CL1, Para 3 to Page 821, CL1, Para 1), because recovery times depend on the checkpointing strategy such as number of transactions completed between the checkpoints (Abstract, L3-12); and the objective is to minimize the time spent in checkpointing and recovery; increasing the frequency of checkpointing decreases recovery time but increases the checkpointing time; decreasing the frequency of checkpointing decreases the checkpointing time but increases the recovery time; there is an optimum checkpointing strategy which maximizes the system availability for processing new transactions and minimizes the mean response time of a transaction (Page 807, CL2, Para 1, L12-23). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to

modify the method of **Lee et al.** with the method of **Nicola et al.** that included the dirty buffer limit being determined from the simulated MTTR setting and historical operating data. The artisan would have been motivated because recovery times would depend on the checkpointing strategy such as number of transactions completed between the checkpoints; and the objective would be to minimize the time spent in checkpointing and recovery; increasing the frequency of checkpointing would decrease recovery time but would increase the checkpointing time; decreasing the frequency of checkpointing would decrease the checkpointing time but would increase the recovery time; there would be an optimum checkpointing strategy which would maximize the system availability for processing new transactions and would minimize the mean response time of a transaction.

9.3 As per Claims 13 and 18, these are rejected based on the same reasoning as Claims 3 and 8, supra. Claims 13 and 18 are computer-readable storage medium claims reciting the same limitations as Claims 3 and 8, as taught throughout by **Lee et al., Nicola et al., Joshi et al.** and **Schofield et al.**

9.4 As per claim 27, **Lee et al., Nicola et al.** and **Joshi et al.** teach the system of claim 24. **Lee et al.** does not expressly teach the predetermined length being a dirty buffer limit. **Joshi et al.** teaches a dirty buffer where data structures used for checkpointing are kept in the cache (Page 667, CL2, Para 3), because that allows checkpointing schemes to write buffers from the head of the checkpoint queue, so the instance of checkpoint keeps advancing (Page 667, CL2, Para 3). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to

modify the system of **Lee et al.** with the system of **Joshi et al.** that included a dirty buffer where data structures used for checkpointing were kept in the cache. The artisan would have been motivated because that would allow checkpointing schemes to write buffers from the head of the checkpoint queue, so the instance of checkpoint would keep advancing.

Schofield et al. teaches the predetermined length being a dirty buffer limit (CL7, L60-64), because that allows writing the buffers and the logs to the disk (nonvolatile memory) when the information in the virtual storage reaches a predetermined size (CL7, L60-64). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the system of **Lee et al.** with the system of **Schofield et al.** that included the predetermined length being a dirty buffer limit. The artisan would have been motivated because that would allow writing the buffers and the logs to the disk (nonvolatile memory) when the information in the virtual storage reached a predetermined size.

Lee et al. does not expressly teach the dirty buffer limit being determined from the simulated MTTR setting and historical operating data. **Nicola et al.** teaches the dirty buffer limit being determined from the simulated MTTR setting and historical operating data (Abstract, L3-8 and L11-12; Page 818, CL2, Para 2, L1-2; Page 820, CL1, Para 3 to Page 821, CL1, Para 1), because recovery times depend on the checkpointing strategy such as number of transactions completed between the checkpoints (Abstract, L3-12); and the objective is to minimize the time spent in checkpointing and recovery; increasing the frequency of checkpointing decreases recovery time but increases the checkpointing time; decreasing the frequency of checkpointing decreases the checkpointing time but increases the recovery time; there is an optimum checkpointing strategy which maximizes the system availability for processing new transactions

and minimizes the mean response time of a transaction (Page 807, CL2, Para 1, L12-23). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the system of **Lee et al.** with the system of **Nicola et al.** that included the dirty buffer limit being determined from the simulated MTTR setting and historical operating data. The artisan would have been motivated because recovery times would depend on the checkpointing strategy such as number of transactions completed between the checkpoints; and the objective would be to minimize the time spent in checkpointing and recovery; increasing the frequency of checkpointing would decrease recovery time but would increase the checkpointing time; decreasing the frequency of checkpointing would decrease the checkpointing time but would increase the recovery time; there would be an optimum checkpointing strategy which would maximize the system availability for processing new transactions and would minimize the mean response time of a transaction.

10. Claims 4-7 and 14-17 are rejected under 35 U.S.C. 103(a) as being unpatentable over **Lee et al.** ("Checkpointing schemes for fast restart in main memory database system", IEEE, 1997) in view of **Nicola et al.** ("Comparative analysis of different models of checkpointing and recovery", IEEE 1990), and further in view of **Joshi et al.** ("Checkpointing in Oracle", Proceedings of the 24th VLDB conference, 1998) and **Frank et al.** (U.S. Patent 5,996,088).

10.1 As per claim 4, **Lee et al., Nicola et al.** and **Joshi et al.** teach the method of claim 1. **Lee et al.** does not expressly teach in response to detecting a write out of a second buffer from volatile memory and storing in nonvolatile memory, checking if the second buffer is represented

in the simulated checkpoint queue. **Joshi et al.** teaches in response to detecting a write out of a second buffer from volatile memory and storing in nonvolatile memory, checking if the second buffer is represented in the simulated checkpoint queue (Page 666, CL1, Para 3 to CL2, Para 1; Page 666, CL2, Para 2 and 3), because as per **Frank et al.** checkpointing periodically updates the data stored in the database with the committed data stored in the volatile cache memory; and by checkpointing, the database is kept relatively up to date, so that when system failure occurs, less recovery needs to be done (CL1, L63-67). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the method of **Lee et al.** with the method of **Joshi et al.** that included in response to detecting a write out of a second buffer from volatile memory and storing in nonvolatile memory, checking if the second buffer is represented in the simulated checkpoint queue. The artisan would have been motivated because checkpointing periodically would update the data stored in the database with the committed data stored in the volatile cache memory; and by checkpointing, the database would be kept relatively up to date, so that when system failure occurred, less recovery would need to be done.

Lee et al. does not expressly teach if the second buffer is represented in the simulated checkpoint queue, removing the element representing the second buffer from the simulated checkpoint queue and incrementing the simulated write counter. **Joshi et al.** teaches if the second buffer is represented in the simulated checkpoint queue, removing the element representing the second buffer from the simulated checkpoint queue and incrementing the simulated write counter (Page 666, CL2, Para 3 to Page 667, CL1, Para 1), because that allows checkpointing to record the checkpoint as having completed and update the checkpoint progress record (redo log) (Page 667, CL1, Para 1). It would have been obvious to one of ordinary skill in

the art at the time of Applicants' invention to modify the method of **Lee et al.** with the method of **Joshi et al.** that included if the second buffer was represented in the simulated checkpoint queue, removing the element representing the second buffer from the simulated checkpoint queue and incrementing the simulated write counter. The artisan would have been motivated because that would allow checkpointing to record the checkpoint as having completed and update the checkpoint progress record (redo log).

10.2 As per claim 5, **Lee et al., Nicola et al., Joshi et al.** and **Frank et al.** teach the method of claim 4. **Lee et al.** teaches the write out of the second buffer being caused by an incremental checkpoint operation (Page 663, CL1, Para 3, L6-8).

10.3 As per claim 6, **Lee et al., Nicola et al.** and **Joshi et al.** teach the method of claim 1. **Lee et al.** does not expressly teach that each element in the simulated checkpoint queue comprises a first identifier that identifies an associated buffer. **Joshi et al.** teaches that each element in the simulated checkpoint queue comprises a first identifier that identifies an associated buffer (Page 665, CL2, Para 3), because checkpointing organizes all modified buffers on ordered queues, to increase efficiency in identifying the precise set of buffers that need to be written for checkpoints (Page 665, CL2, Para 3). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the method of **Lee et al.** with the method of **Joshi et al.** that included each element in the simulated checkpoint queue comprising a first identifier that identifies an associated buffer. The artisan would have been motivated because checkpointing

would organize all modified buffers on ordered queues, to increase efficiency in identifying the precise set of buffers that would need to be written for checkpoints.

Lee et al. does not expressly teach that each element in the simulated checkpoint queue comprises a second identifier that identifies a journal entry in a redo log, the journal entry corresponding to the associated buffer. **Frank et al.** teaches that each element in the simulated checkpoint queue comprises a second identifier that identifies a journal entry in a redo log, the journal entry corresponding to the associated buffer (CL4, L67 to CL5, L9; CL7, L28-35), because that allows a copy of all changes to the records that were made subsequent to the latest checkpoint to be stored in the log file (CL3, L24-44). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the method of **Lee et al.** with the method of **Frank et al.** that included each element in the simulated checkpoint queue comprising a second identifier that identified a journal entry in a redo log, the journal entry corresponding to the associated buffer. The artisan would have been motivated because that would allow a copy of all changes to the records that were made subsequent to the latest checkpoint to be stored in the log file.

10.4 As per claim 7, **Lee et al., Nicola et al., Joshi et al.** and **Frank et al.** teach the method of claim 6. **Lee et al.** does not expressly teach that the elements in the simulated checkpoint queue are ordered. **Joshi et al.** teaches that the elements in the simulated checkpoint queue are ordered (Page 665, CL2, Para 3), because that increases efficiency in identifying the precise set of buffers that need to be written for checkpoints (Page 665, CL2, Para 3). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the method of **Lee**

et al. with the method of **Joshi et al.** that included the elements in the simulated checkpoint queue being ordered. The artisan would have been motivated because that would increase efficiency in identifying the precise set of buffers that would need to be written for checkpoints.

Lee et al. does not expressly teach that the elements in the simulated checkpoint queue are ordered according to each element's journal entry position in the redo log. **Frank et al.** teaches that the elements in the simulated checkpoint queue are ordered according to each element's journal entry position in the redo log (CL4, L67 to CL5, L9; CL7, L28-35), because that allows a copy of all changes to the records that were made subsequent to the latest checkpoint to be stored in the log file (CL3, L24-44); and in case of failure, allows the log files to be accessed to update the main memory to reflect any changes that were made between the time of the latest checkpoint until the time of failure (CL3, L52-55). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the method of **Lee et al.** with the method of **Frank et al.** that included the elements in the simulated checkpoint queue being ordered according to each element's journal entry position in the redo log. The artisan would have been motivated because that would allow a copy of all changes to the records that were made subsequent to the latest checkpoint to be stored in the log file; and in case of failure, would allow the log files to be accessed to update the main memory to reflect any changes that were made between the time of the latest checkpoint until the time of failure.

10.5 As per Claims 14-17, these are rejected based on the same reasoning as Claims 4-7, supra. Claims 14-17 are computer-readable storage medium claims reciting the same limitations as Claims 4-7, as taught throughout by **Lee et al., Nicola et al., Joshi et al.** and **Frank et al.**

11. Claims 9-10, 19-20 and 28-29 are rejected under 35 U.S.C. 103(a) as being unpatentable over **Lee et al.** ("Checkpointing schemes for fast restart in main memory database system", IEEE, 1997) in view of **Nicola et al.** ("Comparative analysis of different models of checkpointing and recovery", IEEE 1990), and **Joshi et al.** ("Checkpointing in Oracle", Proceedings of the 24th VLDB conference, 1998), and further in view of **Schofield et al.** (U.S. Patent 6,493,826) and **Frank et al.** (U.S. Patent 5,996,088).

11.1 As per claims 9 and 10, **Lee et al., Nicola et al., Joshi et al.** and **Schofield et al.** teach the method of claim 8. **Lee et al.** does not expressly teach that the historical operating data comprises an average time to read one journal entry in a redo log; and the historical operating data comprises an average time to read one buffer from nonvolatile memory to volatile memory. **Frank et al.** teaches that the historical operating data comprises an average time to read one journal entry in a redo log; and the historical operating data comprises an average time to read one buffer from nonvolatile memory to volatile memory (CL1, L56-59; CL2, L28-31), because upon recovery after a failure, the data stored in the database on the nonvolatile disk is read from the disk and stored back in the memory (CL2, L25-31); and the goal is to recover from failure as quickly as possible by minimizing the time required to bring the system back on-line (CL1, L56-59). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the method of **Lee et al.** with the method of **Frank et al.** that included the historical operating data comprising an average time to read one journal entry in a redo log; and the historical operating data comprising an average time to read one buffer from nonvolatile

memory to volatile memory. The artisan would have been motivated because upon recovery after a failure, the data stored in the database on the nonvolatile disk would be read from the disk and stored back in the memory; and the goal would be to recover from failure as quickly as possible by minimizing the time required to bring the system back on-line.

11.2 As per Claims 19-20, these are rejected based on the same reasoning as Claims 9-10, supra. Claims 19-20 are computer-readable storage medium claims reciting the same limitations as Claims 9-10, as taught throughout by **Lee et al., Nicola et al., Joshi et al., Schofield et al.** and **Frank et al.**

11.3 As per claims 28 and 29, **Lee et al., Nicola et al., Joshi et al.** and **Schofield et al.** teach the system of claim 27. **Lee et al.** does not expressly teach that the historical operating data comprises an average time to read one journal entry in a redo log; and the historical operating data comprises an average time to read one buffer from nonvolatile memory to volatile memory. **Frank et al.** teaches that the historical operating data comprises an average time to read one journal entry in a redo log; and the historical operating data comprises an average time to read one buffer from nonvolatile memory to volatile memory (CL1, L56-59; CL2, L28-31), because upon recovery after a failure, the data stored in the database on the nonvolatile disk is read from the disk and stored back in the memory (CL2, L25-31); and the goal is to recover from failure as quickly as possible by minimizing the time required to bring the system back on-line (CL1, L56-59). It would have been obvious to one of ordinary skill in the art at the time of Applicants' invention to modify the system of **Lee et al.** with the system of **Frank et al.** that included the historical operating data comprising an average time to read one journal entry in a redo log; and

the historical operating data comprising an average time to read one buffer from nonvolatile memory to volatile memory. The artisan would have been motivated because upon recovery after a failure, the data stored in the database on the nonvolatile disk would be read from the disk and stored back in the memory; and the goal would be to recover from failure as quickly as possible by minimizing the time required to bring the system back on-line.

Conclusion

12. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Dr. Kandasamy Thangavelu whose telephone number is 571-272-3717. The examiner can normally be reached on Monday through Friday from 8:00 AM to 5:30 PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kevin Teska, can be reached on 571-272-3716. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to TC 2100 Group receptionist: 571-272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR.

Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

K. Thangavelu
Art Unit 2123
March 17, 2005